REMARKS

The Office Action basically utilized the principal reference of *Eldering* (U.S. Patent No. 6,615,039) to disclose a broadcasting method for reducing television receiver latencies in displaying an interactive content portion of a broadcast television commercial. The Office Action maintained that the program maps taught by *Eldering* could be broadly interpreted as providing instructions to STB and that the difference between an MPEG-compliant program map of *Eldering* and our claim terminology of "scripts" would be the same.

While a PMT in *Eldering* is repeatedly transmitted in a predetermined time period, the PMT, however, is a table associated with PIDs, packet IDs and programs. It does not instruct any similar control of a receiving apparatus as set forth in our claims.

To address this issue, applicant has amended the claims to clarify the significant differences. Accordingly, our claims define a receiving apparatus based upon event messages which allow for playback of data broadcasting at the start of CM, and the data broadcasting contents are multiplexed with the main program prior to the start of the CM.

The *Eldering* reference, does not through the PMTs, provide any teaching of a transmission unit for generating an instruction as a script and transmitting the script to the receiving apparatus together with the transmission of the advertisement. The script can be multiplexed with the mobile program as the data broadcasting contents based on a data carousel transmission method and transmitted by a transmission unit from the broadcasting apparatus to the receiving apparatus. The receiving apparatus can, accordingly, store and timely respond as a result of event messages that can respectively instruct storage or reproduction.

The Office Action supplemented the teaching of the *Eldering* reference to reject Claim 24 over a combination of the *Eldering* reference in view of *Swix et al.* (U.S. Patent No. 6,718,551).

Claims 1, 2, 4, 9, 11 and 12 and 14-23 were also rejected over a combination of *Eldering* in view of *Suzuki* (U.S. Patent No. 6,401,243), when taken in view of the *Swix et al.* disclosure.

In the specification of the present application, an explanation is given for using the term "message," instead of the term "script."

For example, as described on Page 17, Line 15 through Page 19, Line 13 in our specification, the script (message) of the present invention is included in a data module with ID=0 in each content, and is depicted in Figure 5 as messages M1-M5.

Note that, as amended in the current claims, the script (message) is automatically stored when received by the receiving apparatus (see "An ID is assigned to a data module, and the data module with the ID=0 <u>must be cached in the buffer by the receiving apparatus even though there is no explicit instruction</u>." Page 17, Lines 5-7 and "The data module decode unit 208 decodes the data module with ID=0 in the data stream to reconstruct contents data, and when receiving the instruction to cache the contents data with the specified ID from the contents data processing unit 210, decodes the data module with the specified ID." Page 27, Lines 20-24.

An event message is transmitted to the receiving apparatus at a predetermined timing based on an event message transmission schedule (see Page 19, Lines 14 through Page 20, Line 6 in the specification, and the instruction shown in Figure 5).

With the event message, scripts having been stored in the receiving apparatus are executed to thereby cache target contents in the buffer, or reproduce target contents.

The messages M1-M5 included in the data module with ID=0 cached in the receiving apparatus as above operate to, when the event message is received, cache a data module indicated by the event message. A person of ordinary skill in the art would readily understand that the messages M1-M5 correspond to so-called "scripts" and certainly would understand the difference between an MPEG program map.

The "program map" of *Eldering* is different from the "script" of the present invention, as used for instructing caching and reproduction, and does not have a similar function as a script.

A supplementary reference to a "program map" as known in the art can be seen in the ISO/IEC 13818-1 Section 2.4.4 (copy attached), which is the table specification of PMT (Program Map Table), referred to by *Eldering*. It is requested that this evidence be made of record in this presentation, *In re Sullivan et al.* 498 F.3d 1345 (Fed. Cir. 2007).

As can be seen from this reference, a PMT is just a table associating PIDs (packet IDs) and programs, and *Eldering* does not disclose any description corresponding to the "script" of the present invention and a PMT is not directed to provide corresponding instructions for the control of the receiving apparatus.

Eldering, in Column 11, Lines 1-21, discloses that, when a receiver inserts into programs advertisements pertaining to "automobiles" according to the user's preference, car advertisements (auxiliary data) are retrieved in advance to the receiving apparatus by specifying PIDs of packets made up of the advertisements using a table of the program map.

The present invention has a structure in which scripts (messages) are transmitted in advance from the transmitting apparatus to the receiving apparatus, and an event message is transmitted to the receiving apparatus to cache transmitted contents, and causes the cached contents to be reproduced. *Eldering* neither discloses nor indicates such a structure nor effects unique to the present invention.

New Claim 29 has been added and recites that, instead of being multiplexed onto contents as a content of the data module, the "scripts" are repeatedly transmitted as event messages, independently of the contents.

Herewith, since there are no messages depending on another program in the data module, it is possible to realize an effect that permits reuse of the data module (e.g., rebroadcasting of the program becomes facilitated (see Page 36, Lines 11-17 of the specification).

The *Swix* reference was cited for teaching a transmission of data in a carousel format, citing Column 9, Lines 32-44. Basically, this citation refers to a set-top box that could seek to retrieve targeted advertisements based on a demographic group, apparently to be inserted in an appropriate time. The broadcast carousel format apparently refers to bitmap advertisements that could be spooled in a broadcast carousel format. Needless to say, the *Swix* reference does not address the issues raised with regards to the deficiencies in the *Eldering* disclosure.

The *Suzuki* reference is cited to teach a (script) set of instructions that were generated to reproduce the program data of a specific program. The program data was stored in a storage unit. More specifically, when a reproduction start control signal is received from a cable television station, a temporary memory device could read corresponding digital video data for a program, and supply it to a digital television decoding circuit through a switch circuit 318 in Figure 8.

Again, it is respectfully submitted that the *Suzuki* reference does not address the deficiencies in *Eldering* and the Office Action has taken a broad interpretation of scripts. Accordingly, applicant has amended the current claims to provide an appropriate distinction.

With such clarification, it is believed the present application is allowable and is not obvious over any combination of the references of record.

> It is the Examiner's burden to establish *prima facie* obviousness. *See In re Rijckaert*, 9 F.3d 1531, 1532 (Fed. Cir. 1993) Obviousness requires a suggestion of all the elements in a claim *(CFMT, Inc. v. Yieldup Int'l Corp.,* 349 F.3d 1333, 1342 (Fed. Cir. 2003)) and "a reason that would have prompted a person of ordinary skill in the relevant field to combine the elements in the way the claimed new invention does." *KSR Int'l Co. v. Teleflex Inc.*, 127 S. Ct. 1727, 1741 (2007). Here, we find that the

> Examiner has not identified all the elements of claim 1, nor provided a reason that would have prompted the skilled worker to have arranged them in the manner necessary to reach the claimed invention.
>
> *Ex parte* Karoleen B. Alexander, No. 2007-2698, slip op. at 6 (B.P.A.I. Nov. 30, 2007)

Applicant believes that the case is now in condition for allowance and an early notification of the same is requested.

If the Examiner believes a telephone interview will help further the prosecution of this case, the undersigned attorney can be contacted at the listed telephone number.

Very truly yours,

SNELL & WILMER L.L.P.

Joseph W. Price
Registration No. 25,124
600 Anton Boulevard, Suite 1400
Costa Mesa, California 92626-7689
Telephone: (714) 427-7420
Facsimile: (714) 427-7799

INTERNATIONAL STANDARD

ISO/IEC 13818-1

Second edition
2000-12-01

Information technology — Generic coding of moving pictures and associated audio information: Systems

Technologies de l'information — Codage générique des images animées et du son associé: Systèmes

Reference number
ISO/IEC 13818-1:2000(E)

Contents

When reconstructing the Program Stream at the Transport Stream decoder, for PES packets with a stream_id value of ancillary_stream_id, packet_start_code_prefix is written to the Program Stream being reconstructed, followed by the data_byte fields from these Transport Stream PES packets.

ISO/IEC 11172-1 streams are carried within Transport Streams by first replacing ISO/IEC 11172-1 packet headers with ITU-T Rec. H.262 | ISO/IEC 13818-2 PES packet headers. ISO/IEC 11172-1 packet header field values are copied to the equivalent ITU-T Rec. H.262 | ISO/IEC 13818-2 PES packet header fields.

The program_packet_sequence_counter field is included within the header of each PES packet carrying data from a Program Stream, or an ISO/IEC 11172-1 System stream. This allows the order of PES packets in the original Program Stream, or packets in the original ISO/IEC 11172-1 System stream, to be reproduced at the decoder.

The pack_header() field of a Program Stream, or an ISO/IEC 11172-1 System stream, is carried in the Transport Stream in the header of the immediately following PES packet.

2.4.4 Program specific information

Program Specific Information (PSI) includes both ITU-T Rec. H.222.0 | ISO/IEC 13818-1 normative data and private data that enable demultiplexing of programs by decoders. Programs are composed of one or more elementary streams, each labelled with a PID. Programs, elementary streams or parts thereof may be scrambled for conditional access. However, Program Specific Information shall not be scrambled.

In Transport Streams, Program Specific Information is classified into five table structures as shown in Table 2-23. While these structures may be thought of as simple tables, they shall be segmented into sections and inserted in Transport Stream packets, some with predetermined PIDs and others with user selectable PIDs.

Table 2-23 – Program specific information

Structure Name	Stream Type	PID number	Description
Program Association Table	ITU-T Rec. H.222.0 | ISO/IEC 13818-1	0x00	Associates Program Number and Program Map Table PID
Program Map Table	ITU-T Rec. H.222.0 | ISO/IEC 13818-1	Assignment indicated in the PAT	Specifies PID values for components of one or more programs
Network Information Table	Private	Assignment indicated in the PAT	Physical network parameters such as FDM frequencies, Transponder Numbers, etc.
Conditional Access Table	ITU-T Rec. H.222.0 | ISO/IEC 13818-1	0x01	Associates one or more (private) EMM streams each with a unique PID value
Transport Stream Description Table	ITU-T Rec. H.222.0 | ISO/IEC 13818-1	0x02	Associates one or more descriptors from Table 2-39 to an entire Transport Stream

ITU-T Rec. H.222.0 | ISO/IEC 13818-1 defined PSI tables shall be segmented into one or more sections that are carried within transports packets. A section is a syntactic structure that shall be used for mapping each ITU-T Rec. H.222.0 | ISO/IEC 13818-1 defined PSI table into Transport Stream packets.

Along with ITU-T Rec. H.222.0 | ISO/IEC 13818-1 defined PSI tables, it is possible to carry private data tables. The means by which private information is carried within Transport Stream packets is not defined by this Specification. It may be structured in the same manner used for carrying of ITU-T Rec. H.222.0 | ISO/IEC 13818-1 defined PSI tables, such that the syntax for mapping this private data is identical to that used for the mapping of ITU-T Rec. H.222.0 | ISO/IEC 13818-1 defined PSI tables. For this purpose, a private section is defined. If the private data is carried in Transport Stream packets with the same PID value as Transport Stream packets carrying Program Map Tables, (as identified in the Program Association Table), then the private_section syntax and semantics shall be used. The data carried in the private_data_bytes may be scrambled. However, no other fields of the private_section shall be scrambled. This private_section allows data to be transmitted with a minimum of structure. When this structure is not used, the mapping of private data within Transport Stream packets is not defined by this Recommendation | International Standard.

Sections may be variable in length. The beginning of a section is indicated by a pointer_field in the Transport Stream packet payload. The syntax of this field is specified in Table 2-24.

Adaptation fields may occur in Transport Stream packets carrying PSI sections.

Within a Transport Stream, packet stuffing bytes of value 0xFF may be found in the payload of Transport Stream packets carrying PSI and/or private_sections only after the last byte of a section. In this case all bytes until the end of the Transport Stream packet shall also be stuffing bytes of value 0xFF. These bytes may be discarded by a decoder. In such a case, the payload of the next Transport Stream packet with the same PID value shall begin with a pointer_field of value 0x00 indicating that the next section starts immediately thereafter.

Each Transport Stream shall contain one or more Transport Stream packets with PID value 0x0000. These Transport Stream packets together shall contain a complete Program Association Table, providing a complete list of all programs within the Transport Stream. The most recently transmitted version of the table with the current_next_indicator set to a value of '1' shall always apply to the current data in the Transport Stream. Any changes in the programs carried within the Transport Stream shall be described in an updated version of the Program Association Table carried in Transport Stream packets with PID value 0x0000. These sections shall all use table_id value 0x00. Only sections with this value of table_id are permitted within Transport Stream packets with PID value of 0x0000. For a new version of the PAT to become valid, all sections (as indicated in the last_section_number) with a new version_number and with the current_next_indicator set to '1' must exit B_{sys} defined in the T-STD (refer to 2.4.2). The PAT becomes valid when the last byte of the section needed to complete the table exits B_{sys}.

Whenever one or more elementary streams within a Transport Stream are scrambled, Transport Stream packets with a PID value 0x0001 shall be transmitted containing a complete Conditional Access Table including CA_descriptors associated with the scrambled streams. The transmitted Transport Stream packets will together form one complete version of the conditional access table. The most recently transmitted version of the table with the current_next_indicator set to a value of '1' shall always apply to the current data in the Transport Stream. Any changes in scrambling making the existing table invalid or incomplete shall be described in an updated version of the conditional access table. These sections will all use table_id value 0x01. Only sections with this table_id value are permitted within Transport Stream packets with a PID value of 0x0001. For a new version of the CAT to become valid, all sections (as indicated in the last_section_number) with a new version_number and with the current_next_indicator set to '1' must exit B_{sys}. The CAT becomes valid when the last byte of the section needed to complete the table exits B_{sys}.

Each Transport Stream shall contain one or more Transport Stream packets with PID values which are labelled under the program association table as Transport Stream packets containing TS program map sections. Each program listed in the Program Association Table shall be described in a unique TS_program_map_section. Every program shall be fully defined within the Transport Stream itself. Private data which has an associated elementary_PID field in the appropriate Program Map Table section is part of the program. Other private data may exist in the Transport Stream without being listed in the Program Map Table section. The most recently transmitted version of the TS_program_map_section with the current_next_indicator set to a value of '1' shall always apply to the current data within the Transport Stream. Any changes in the definition of any of the programs carried within the Transport Stream shall be described in an updated version of the corresponding section of the program map table carried in Transport Stream packets with the PID value identified as the program_map_PID for that specific program. All Transport Stream packets which carry a given TS_program_map_section shall have the same PID value. During the continuous existence of a program, including all of its associated events, the program_map_PID shall not change. A program definition shall not span more than one TS_program_map_section. A new version of a TS_program_map_section becomes valid when the last byte of that section with a new version_number and with the current_next_indicator set to '1' exits B_{sys}.

Sections with a table_id value of 0x02 shall contain Program Map Table information. Such sections may be carried in Transport Stream packets with different PID values.

The Network Information Table is optional and its contents are private. If present it is carried within Transport Stream packets that will have the same PID value, called the network_PID. The network_PID value is defined by the user and, when present, shall be found in the Program Association Table under the reserved program_number 0x0000. If the network information table exists, it shall take the form of one or more private_sections.

The maximum number of bytes in a section of a ITU-T Rec. H.222.0 | ISO/IEC 13818-1 defined PSI table is 1024 bytes. The maximum number of bytes in a private_section is 4096 bytes.

The Transport Stream Description Table is optional. When present, the Transport Stream Description is carried within Transport Stream packets that have a PID value 0x0002 as specified in Table 2-23 and shall apply to the entire Transport Stream. Sections of the Transport Stream Description shall use a table_id value of 0x03 as specified in Table 2-26 and its contents are restricted to descriptors specified in Table 2-39. The TS_description_section becomes valid when the last byte of the section required to complete the table exits B_{sys}.

There are no restrictions on the occurrence of start codes, sync bytes or other bit patterns in PSI data, whether this Recommendation | International Standard or private.

2.4.4.1 Pointer

The pointer_field syntax is defined in Table 2-24.

Table 2-24 – Program specific information pointer

Syntax	No. of bits	Mnemonic
pointer_field	**8**	**uimsbf**

2.4.4.2 Semantics definition of fields in pointer syntax

pointer_field – This is an 8-bit field whose value shall be the number of bytes, immediately following the pointer_field until the first byte of the first section that is present in the payload of the Transport Stream packet (so a value of 0x00 in the pointer_field indicates that the section starts immediately after the pointer_field). When at least one section begins in a given Transport Stream packet, then the payload_unit_start_indicator (refer to 2.4.3.2) shall be set to 1 and the first byte of the payload of that Transport Stream packet shall contain the pointer. When no section begins in a given Transport Stream packet, then the payload_unit_start_indicator shall be set to 0 and no pointer shall be sent in the payload of that packet.

2.4.4.3 Program association Table

The Program Association Table provides the correspondence between a program_number and the PID value of the Transport Stream packets which carry the program definition. The program_number is the numeric label associated with a program.

The overall table is contained in one or more sections with the following syntax. It may be segmented to occupy multiple sections (see Table 2.25).

Table 2-25 – Program association section

Syntax	No. of bits	Mnemonic
program_association_section() {		
table_id	**8**	**uimsbf**
section_syntax_indicator	**1**	**bslbf**
'0'	**1**	**bslbf**
reserved	**2**	**bslbf**
section_length	**12**	**uimsbf**
transport_stream_id	**16**	**uimsbf**
reserved	**2**	**bslbf**
version_number	**5**	**uimsbf**
current_next_indicator	**1**	**bslbf**
section_number	**8**	**uimsbf**
last_section_number	**8**	**uimsbf**
for (i = 0; i < N; i++) {		
program_number	**16**	**uimsbf**
reserved	**3**	**bslbf**
if (program_number == '0') {		
network_PID	**13**	**uimsbf**
}		
else {		
program_map_PID	**13**	**uimsbf**
}		
}		
CRC_32	**32**	**rpchof**
}		

2.4.4.4 Table_id assignments

The table_id field identifies the contents of a Transport Stream PSI section as shown in Table 2-26.

Table 2-26 – table_id assignment values

Value	description
0x00	program_association_section
0x01	conditional_access_section (CA_section)
0x02	TS_program_map_section
0x03	TS_description_section
0x04	ISO_IEC_14496_scene_description_section
0x05	ISO_IEC_14496_object_descriptor_section
0x06-0x37	ITU-T Rec. H.222.0 | ISO/IEC 13818-1 reserved
0x38-0x3F	Defined in ISO/IEC 13818-6
0x40-0xFE	User private
0xFF	forbidden

2.4.4.5 Semantic definition of fields in program association section

table_id – This is an 8-bit field, which shall be set to 0x00 as shown in Table 2-26.

section_syntax_indicator – The section_syntax_indicator is a 1-bit field which shall be set to '1'.

section_length – This is a 12-bit field, the first two bits of which shall be '00'. The remaining 10 bits specify the number of bytes of the section, starting immediately following the section_length field, and including the CRC. The value in this field shall not exceed 1021 (0x3FD).

transport_stream_id – This is a 16-bit field which serves as a label to identify this Transport Stream from any other multiplex within a network. Its value is defined by the user.

version_number – This 5-bit field is the version number of the whole Program Association Table. The version number shall be incremented by 1 modulo 32 whenever the definition of the Program Association Table changes. When the current_next_indicator is set to '1', then the version_number shall be that of the currently applicable Program Association Table. When the current_next_indicator is set to '0', then the version_number shall be that of the next applicable Program Association Table.

current_next_indicator – A 1-bit indicator, which when set to '1' indicates that the Program Association Table sent is currently applicable. When the bit is set to '0', it indicates that the table sent is not yet applicable and shall be the next table to become valid.

section_number – This 8-bit field gives the number of this section. The section_number of the first section in the Program Association Table shall be 0x00. It shall be incremented by 1 with each additional section in the Program Association Table.

last_section_number – This 8-bit field specifies the number of the last section (that is, the section with the highest section_number) of the complete Program Association Table.

program_number – Program_number is a 16-bit field. It specifies the program to which the program_map_PID is applicable. When set to 0x0000, then the following PID reference shall be the network PID. For all other cases the value of this field is user defined. This field shall not take any single value more than once within one version of the Program Association Table.

NOTE – The program_number may be used as a designation for a broadcast channel, for example.

network_PID – The network_PID is a 13-bit field, which is used only in conjunction with the value of the program_number set to 0x0000, specifies the PID of the Transport Stream packets which shall contain the Network Information Table. The value of the network_PID field is defined by the user, but shall only take values as specified in Table 2-3. The presence of the network_PID is optional.

program_map_PID – The program_map_PID is a 13-bit field specifying the PID of the Transport Stream packets which shall contain the program_map_section applicable for the program as specified by the program_number. No program_number shall have more than one program_map_PID assignment. The value of the program_map_PID is defined by the user, but shall only take values as specified in Table 2-3.

CRC_32 – This is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in Annex A after processing the entire program association section.

2.4.4.6 Conditional access Table

The Conditional Access (CA) Table provides the association between one or more CA systems, their EMM streams and any special parameters associated with them. Refer to 2.6.16 for a definition of the descriptor() field in Table 2-27.

The table is contained in one or more sections with the following syntax. It may be segmented to occupy multiple sections.

Table 2-27 – Conditional access section

Syntax	No. of bits	Mnemonic
CA_section() {		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
'0'	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
reserved	18	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
for (i = 0; i < N; i++) {		
descriptor()		
}		
CRC_32	32	rpchof
}		

2.4.4.7 Semantic definition of fields in conditional access section

table_id – This is an 8-bit field, which shall be set to 0x01 as specified in Table 2-26.

section_syntax_indicator – The section_syntax_indicator is a 1-bit field which shall be set to '1'.

section_length – This is a 12-bit field, the first two bits of which shall be '00'. The remaining 10-bits specify the number of bytes of the section starting immediately following the section_length field, and including the CRC. The value in this field shall not exceed 1021 (0x3FD).

version_number – This 5-bit field is the version number of the entire conditional access table. The version number shall be incremented by 1 modulo 32 when a change in the information carried within the CA table occurs. When the current_next_indicator is set to '1', then the version_number shall be that of the currently applicable Conditional Access Table. When the current_next_indicator is set to '0', then the version_number shall be that of the next applicable Conditional Access Table.

current_next_indicator – A 1-bit indicator, which when set to '1' indicates that the Conditional Access Table sent is currently applicable. When the bit is set to '0', it indicates that the Conditional Access Table sent is not yet applicable and shall be the next Conditional Access Table to become valid.

section_number – This 8-bit field gives the number of this section. The section_number of the first section in the Conditional Access Table shall be 0x00. It shall be incremented by 1 with each additional section in the Conditional Access Table.

last_section_number – This 8-bit field specifies the number of the last section (that is, the section with the highest section_number) of the Conditional Access Table.

CRC_32 – This is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in Annex A after processing the entire conditional access section.

2.4.4.8 Program Map Table

The Program Map Table provides the mappings between program numbers and the program elements that comprise them. A single instance of such a mapping is referred to as a "program definition". The program map table is the complete collection of all program definitions for a Transport Stream. This table shall be transmitted in packets, the PID values of which are selected by the encoder. More than one PID value may be used, if desired. The table is contained in one or more sections with the following syntax. It may be segmented to occupy multiple sections. In each section, the section number field shall be set to zero. Sections are identified by the program_number field.

Definition for the descriptor() fields may be found in 2.6 (see Table 2-28).

Table 2-28 – Transport Stream program map section

Syntax	No. of bits	Mnemonic
TS_program_map_section() {		
table_id	8	**uimsbf**
section_syntax_indicator	1	**bslbf**
'0'	1	**bslbf**
reserved	2	**bslbf**
section_length	12	**uimsbf**
program_number	16	**uimsbf**
reserved	2	**bslbf**
version_number	5	**uimsbf**
current_next_indicator	1	**bslbf**
section_number	8	**uimsbf**
last_section_number	8	**uimsbf**
reserved	3	**bslbf**
PCR_PID	13	**uimsbf**
reserved	4	**bslbf**
program_info_length	12	**uimsbf**
for (i = 0; i < N; i++) {		
descriptor()		
}		
for (i = 0; i < N1; i++) {		
stream_type	8	**uimsbf**
reserved	3	**bslbf**
elementary_PID	13	**uimsbf**
reserved	4	**bslbf**
ES_info_length	12	**uimsbf**
for (i = 0; i < N2; i++) {		
descriptor()		
}		
}		
CRC_32	32	**rpchof**
}		

2.4.4.9 Semantic definition of fields in Transport Stream program map section

table_id – This is an 8-bit field, which in the case of a TS_program_map_section shall be always set to 0x02 as shown in Table 2-26.

section_syntax_indicator – The section_syntax_indicator is a 1-bit field which shall be set to '1'.

section_length – This is a 12-bit field, the first two bits of which shall be '00'. The remaining 10 bits specify the number of bytes of the section starting immediately following the section_length field, and including the CRC. The value in this field shall not exceed 1021 (0x3FD).

program_number – program_number is a 16-bit field. It specifies the program to which the program_map_PID is applicable. One program definition shall be carried within only one TS_program_map_section. This implies that a program definition is never longer than 1016 (0x3F8). See Informative Annex C for ways to deal with the cases when that length is not sufficient. The program_number may be used as a designation for a broadcast channel, for example. By describing the different program elements belonging to a program, data from different sources (e.g. sequential events) can be concatenated together to form a continuous set of streams using a program_number. For examples of applications refer to Annex C.

version_number – This 5-bit field is the version number of the TS_program_map_section. The version number shall be incremented by 1 modulo 32 when a change in the information carried within the section occurs. Version number refers to the definition of a single program, and therefore to a single section. When the current_next_indicator is set to '1', then the version_number shall be that of the currently applicable TS_program_map_section. When the current_next_indicator is set to '0', then the version_number shall be that of the next applicable TS_program_map_section.

current_next_indicator – A 1-bit field, which when set to '1' indicates that the TS_program_map_section sent is currently applicable. When the bit is set to '0', it indicates that the TS_program_map_section sent is not yet applicable and shall be the next TS_program_map_section to become valid.

section_number – The value of this 8-bit field shall be 0x00.

last_section_number – The value of this 8-bit field shall be 0x00.

PCR_PID – This is a 13-bit field indicating the PID of the Transport Stream packets which shall contain the PCR fields valid for the program specified by program_number. If no PCR is associated with a program definition for private streams, then this field shall take the value of 0x1FFF. Refer to the semantic definition of PCR in 2.4.3.5 and Table 2-3 for restrictions on the choice of PCR_PID value.

program_info_length – This is a 12-bit field, the first two bits of which shall be '00'. The remaining 10 bits specify the number of bytes of the descriptors immediately following the program_info_length field.

stream_type – This is an 8-bit field specifying the type of program element carried within the packets with the PID whose value is specified by the elementary_PID. The values of stream_type are specified in Table 2-29.

NOTE – An ITU-T Rec. H.222.0 | ISO/IEC 13818-1 auxiliary stream is available for data types defined by this Specification, other than audio, video, and DSM CC, such as Program Stream Directory and Program Stream Map.

elementary_PID – This is a 13-bit field specifying the PID of the Transport Stream packets which carry the associated program element.

ES_info_length – This is a 12-bit field, the first two bits of which shall be '00'. The remaining 10 bits specify the number of bytes of the descriptors of the associated program element immediately following the ES_info_length field.

CRC_32 – This is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in Annex B after processing the entire Transport Stream program map section.

2.4.4.10 Syntax of the Private section

When private data is sent in Transport Stream packets with a PID value designated as a Program Map Table PID in the Program Association Table the private_section shall be used. The private_section allows data to be transmitted with a minimum of structure while enabling a decoder to parse the stream. The sections may be used in two ways: if the section_syntax_indicator is set to '1', then the whole structure common to all tables shall be used; if the indicator is set to '0', then only the fields 'table_id' through 'private_section_length' shall follow the common structure syntax and semantics and the rest of the private_section may take any form the user determines. Examples of extended use of this syntax are found in Informative Annex C.

Table 2-29 – Stream type assignments

Value	Description
0x00	ITU-T | ISO/IEC Reserved
0x01	ISO/IEC 11172 Video
0x02	ITU-T Rec. H.262 | ISO/IEC 13818-2 Video or ISO/IEC 11172-2 constrained parameter video stream
0x03	ISO/IEC 11172 Audio
0x04	ISO/IEC 13818-3 Audio
0x05	ITU-T Rec. H.222.0 | ISO/IEC 13818-1 private_sections
0x06	ITU-T Rec. H.222.0 | ISO/IEC 13818-1 PES packets containing private data
0x07	ISO/IEC 13522 MHEG
0x08	ITU-T Rec. H.222.0 | ISO/IEC 13818-1 Annex A DSM-CC
0x09	ITU-T Rec. H.222.1
0x0A	ISO/IEC 13818-6 type A
0x0B	ISO/IEC 13818-6 type B
0x0C	ISO/IEC 13818-6 type C
0x0D	ISO/IEC 13818-6 type D
0x0E	ITU-T Rec. H.222.0 | ISO/IEC 13818-1 auxiliary
0x0F	ISO/IEC 13818-7 Audio with ADTS transport syntax
0x10	ISO/IEC 14496-2 Visual
0x11	ISO/IEC 14496-3 Audio with the LATM transport syntax as defined in ISO/IEC 14496-3 / AMD 1
0x12	ISO/IEC 14496-1 SL-packetized stream or FlexMux stream carried in PES packets
0x13	ISO/IEC 14496-1 SL-packetized stream or FlexMux stream carried in ISO/IEC14496_sections.
0x14	ISO/IEC 13818-6 Synchronized Download Protocol
0x15-0x7F	ITU-T Rec. H.222.0 | ISO/IEC 13818-1 Reserved
0x80-0xFF	User Private

A private table may be made of several private_sections, all with the same table_id (see Table 2-30).

2.4.4.11 Semantic definition of fields in private section

table_id – This 8-bit field, the value of which identifies the Private Table this section belongs to. Only values defined in Table 2-26 as "user private" may be used.

section_syntax_indicator – This is a 1-bit indicator. When set to '1', it indicates that the private section follows the generic section syntax beyond the private_section_length field. When set to '0', it indicates that the private_data_bytes immediately follow the private_section_length field.

private_indicator – This is a 1-bit user definable flag that shall not be specified by ITU-T | ISO/IEC in the future.

private_section_length – A 12-bit field. It specifies the number of remaining bytes in the private section immediately following the private_section_length field up to the end of the private_section. The value in this field shall not exceed 4093 (0xFFD).

Table 2-30 – Private section

Syntax	No. of bits	Mnemonic
private_section() {		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
private_indicator	1	bslbf
reserved	2	bslbf
private_section_length	12	uimsbf
if (section_syntax_indicator == '0') {		
for (i = 0; i < N; i++) {		
private_data_byte	8	bslbf
}		
}		
else {		
table_id_extension	16	uimsbf
reserved	2	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
for (i = 0; i < private_section_length-9; i++) {		
private_data_byte	8	bslbf
}		
CRC_32	32	rpchof
}		
}		

private_data_byte – The private_data_byte field is user definable and shall not be specified by ITU-T | ISO/IEC in the future.

table_id_extension – This is a 16-bit field. Its use and value are defined by the user.

version_number – This 5-bit field is the version number of the private_section. The version_number shall be incremented by 1 modulo 32 when a change in the information carried within the private_section occurs. When the current_next_indicator is set to '0', then the version_number shall be that of the next applicable private_section with the same table_id and section_number.

current_next_indicator – A 1-bit field, which when set to '1' indicates that the private_section sent is currently applicable. When the current_next_indicator is set to '1', then the version_number shall be that of the currently applicable private_section. When the bit is set to '0', it indicates that the private_section sent is not yet applicable and shall be the next private_section with the same section_number and table_id to become valid.

section_number – This 8-bit field gives the number of the private_section. The section_number of the first section in a private table shall be 0x00. The section_number shall be incremented by 1 with each additional section in this private table.

last_section_number – This 8-bit field specifies the number of the last section (that is, the section with the highest section_number) of the private table of which this section is a part.

CRC_32 – This is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in Annex A after processing the entire private section.

2.4.4.12 Syntax of the Transport Stream section

ITU-T Rec. H.222.0 | ISO/IEC 13818-1 compliant bitstreams may carry the information defined in Table 2-30-1. ITU-T Rec. H.222.0 | ISO/IEC 13818-1 compliant decoders may decode the information defined in this table.

The Transport Stream Description Table is defined to support the carriage of descriptors as found in 2.6 for an entire Transport Stream. The descriptors shall apply to the entire Transport Stream. This table uses a table_id value of 0x03 as specified in Table 2-26 and is carried in Transport Stream packets whose PID value is 0x0002 as specified in Table 2-3.

Table 2-30-1 – The Transport Stream Description Table

Syntax	No. of bits	Mnemonic
TS_description_section() {		
table_id	8	uimsbf
section_syntax_indicator	1	bslbf
'0'	1	bslbf
reserved	2	bslbf
section_length	12	uimsbf
reserved	18	bslbf
version_number	5	uimsbf
current_next_indicator	1	bslbf
section_number	8	uimsbf
last_section_number	8	uimsbf
for (i = 0; i < N; i++) {		
descriptor()		
}		
CRC_32	32	rpchof
}		

2.4.4.13 Semantic definition of fields in the Transport Stream section

table_id – This is an 8-bit field, which shall be set to '0x03' as specified in Table 2-26.

section_length – This is a 12-bit field, the first two bits of which shall be '00'. The remaining 10 bits specify the number of bytes of the section, starting immediately following the section_length field, and including the CRC. The value in this field shall not exceed 1021 (0x3FD).

version_number – This 5-bit field is the version number of the whole Transport Stream Description Table. The version number shall be incremented by 1 modulo 32 whenever the definition of the Transport Stream Description Table changes. When the current_next_indicator is set to '1', then the version_number shall be that of the currently applicable Transport Stream Description Table. When the current_next_indicator is set to '0', then the version_number shall be that of the next applicable Transport Stream Description Table.

current_next_indicator – A 1-bit indicator, which, when set to '1', indicates that the Transport Stream Description Table sent is currently applicable. When the bit is set to '0', it indicates that the table sent is not yet applicable and shall be the next table to become valid.

section_number – This 8-bit field gives the number of this section. The section_number of the first section in the Transport Stream Description Table shall be 0x00. It shall be incremented by 1 with each additional section in the Transport Stream Description Table.

last_section_number – This 8-bit field specifies the number of the last section (that is, the section with the highest section_number) of the complete Transport Stream Description Table.

CRC_32 – This is a 32-bit field that contains the CRC value that gives a zero output of the registers in the decoder defined in Annex A after processing the entire Transport Stream Description section.

2.5 Program Stream bitstream requirements

2.5.1 Program Stream coding structure and parameters

The ITU-T Rec. H.222.0 | ISO/IEC 13818-1 Program Stream coding layer allows one program of one or more elementary streams to be combined into a single stream. Data from each elementary stream are multiplexed together with information that allows synchronized presentation of the elementary streams within the program.

A Program Stream consists of one or more elementary streams from one program multiplexed together. Audio and video elementary streams consist of access units.

Elementary Stream data is carried in PES packets. A PES packet consists of a PES packet header followed by packet data. PES packets are inserted into Program Stream packs.

The PES packet header begins with a 32-bit start-code that also identifies the stream (refer to Table 2-18) to which the packet data belongs. The PES packet header may contain just a Presentation Time Stamp (PTS) or both a presentation timestamp and a Decoding Time Stamp (DTS). The PES packet header also contains other optional fields. The packet data contains a variable number of contiguous bytes from one elementary stream.